REMARKS

Claims 1-14 and 39-40 are pending in the Application. Claims 39-40 are rejected under 35 U.S.C. § 102(b). Claims 1-14 are rejected under 35 U.S.C. § 103(a). Claim 39 is objected to. Applicant respectfully traverses these rejections for at least the reasons stated below and respectfully requests that the Examiner reconsider and withdraw all outstanding rejections.

I. ELECTION/RESTRICTIONS:

Applicant provisionally elects group one, claims 1-14 and 39-40, with traverse. The Restriction Requirement is submitted to be improper for at least the reasons stated by Applicant's attorney, Kelly Kordzik, during a telephone conversation with the Examiner on March 6, 2003.

II. CLAIM OBJECTIONS:

The Examiner has objected to claim 39 for a typographical error. Paper No. 3, page 3. Applicant has amended claim 39 by replacing the phrase "determining if the conditional branch instruction if positioned" with the phrase, "determining if the conditional branch instruction is positioned." Applicant believes that the amendment to claim 39 addresses the Examiner's objection.

III. REJECTIONS UNDER 35 U.S.C. § 102:

The Examiner has rejected claims 39-40 under 35 U.S.C. § 102(b) as being anticipated by *Patterson* and *Hennessy's* Computer Architecture A Quantitative Approach Second Edition (hereinafter "*Hennessy*"). Applicant respectfully traverses these rejections for at least the reasons stated below and respectfully requests that the Examiner reconsider and withdraw these rejections.

For a claim to be anticipated under 35 U.S.C. § 102, each and every claim limitation must be found within the cited prior art reference and arranged as required by the claim. M.P.E.P. §2131.

Hennessy does not disclose "*determining if the conditional branch instruction is positioned at a specified address in a sequence of instructions being executed*" as recited in claim 39. The Examiner directs Applicant's attention to

pages 176-177 and 326 of *Hennessy* as disclosing the above-cited claim limitation. Paper No. 3, page 3. Instead, *Hennessy* discloses:

> A more accurate technique is to predict branches on the basis of profile information collected from earlier runs. The key observation that makes this worthwhile is that the behavior of branches is often bimodally distributed; that is, an individual branch is often highly biased towards taken or on taken. FIGURE 3.36 shows the success of branch prediction using this strategy. The same input data were used for runs and for collecting the profile; other studies have shown that changing the input so that the profile is for a different run leads to only a small change in the accuracy of profile-based prediction. Page 176.
>
> Our data is for several different branch-prediction schemes varying from perfect to no predictor. We assume a separate predictor is used for jumps. Jump predictors are important primarily with the most accurate branch predictors, since branch frequency is higher and the accuracy of the branch predictors dominates. Page 177.

Thus, *Hennessy* discloses *several schemes for predicting branch predictions on the basis of profile information collected from earlier runs.* The Examiner indicated in the interview conducted on July 1, 2003, that the profile information includes address information. However, none of these schemes *determine if a conditional branch instruction is positioned at a specified address.* Further, none of these schemes determine if a conditional branch instruction is positioned at a specified address *in a sequence of instructions being executed. Hennessy* is silent regarding the position of a conditional branch instruction. Thus, *Hennessy* does not disclose all the limitations of claim 39, and thus *Hennessy* does not anticipate claim 39. M.P.E.P. § 2131.

Hennessy does not disclose "*predicting whether the conditional branch instruction will be taken or not taken as a function of the position of the specified address*" as recited in claim 39. The Examiner directs Applicant's attention to pages 176-177 and 326 of *Hennessy* as disclosing the above-cited claim limitation. Paper No. 3, page 3. Instead, as stated above, *Hennessy* discloses several schemes for predicting branch predictions on the basis of profile information collected from earlier runs. However, none of these schemes *determine if a conditional branch instruction is positioned at a specified address.* Further, none of these schemes predict whether the *conditional branch instruction will be taken or not*

taken as a function of the position of a specified address. Thus, *Hennessy* does not disclose all the limitations of claim 39, and thus *Hennessy* does not anticipate claim 39. M.P.E.P. 2131.

For at least the above reasons, claim 39 is not anticipated by *Hennessy.* Claim 40 recites combinations of features including the above combinations, and thus is not anticipated for at least the above reasons as well. Claim 40 recites additional features, which, in combination with the features of the claim upon which it depends, is not anticipated by *Hennessy.*

For example, *Hennessy* does not disclose "wherein the *predicting program step will predict taken if the specified address is a multiple of specified number N*" as recited in claim 40. The Examiner directs Applicant's attention to pages 176-177 and 326 of *Hennessy* as disclosing the above-cited claim limitation. Paper No. 3, page 4. Instead, as stated above, *Hennessy* discloses several schemes for predicting branches on the basis of profile information collected from earlier runs. However, none of these schemes *determine if a conditional branch instruction is positioned at a specified address.* Further, none of these schemes predict a branch taken *if a specified address is a multiple of a specified number N.* Thus, *Hennessy* does not disclose all the limitations of claim 40, and thus *Hennessy* does not anticipate claim 40. M.P.E.P. § 2131.

As a result of the foregoing, Applicant respectfully asserts that not each and every claim limitation is found within *Hennessy*, and thus claims 39-40 are not anticipated by *Hennessy.*

It is noted that limitations are italicized only for emphasis. Limitations that are italicized are not meant to imply that only those limitations are not disclosed in the cited prior art.

IV. REJECTIONS UNDER 35 U.S.C. § 103(a):

The Examiner has rejected claims 1-14 under 35 U.S.C. § 103(a) as being unpatentable over *Burgess* (U.S. Patent No. 6,642,493) in view of Intel's Pentium Processor Family Developer's Manual Volume 3: Architecture and Programming Manual (hereinafter "Intel"). Applicant respectfully traverses these rejections for

at least the reasons stated below and respectfully requests that the Examiner reconsider and withdraw these rejections.

A. The Examiner has not provided any motivation for combining Burgess with Intel

A *prima facie* showing of obviousness requires the Examiner to establish, *inter alia*, that the prior art references teach or suggest, either alone or in combination, all of the limitations of the claimed invention, and the Examiner must provide a motivation or suggestion to combine or modify the prior art reference to make the claimed inventions. M.P.E.P. §2142. The motivation or suggestion to combine references must come from one of three possible sources: the nature of the problem to be solved, the teaching of the prior art and the knowledge of persons of ordinary skill in the art. *In re Rouffet*, 47 U.S.P.Q.2d. 1453,1458 (Fed. Cir. 1998). The showings must be clear and particular. *In re Lee*, 277 F.3d 1338, 1343, 61 U.S.P.Q.2d 1430, 1433-34 (Fed. Cir. 2002); *In re Kotzab*, 217 F.3d 1365, 1370, 55 U.S.P.Q.2d 1313, 1317 (Fed. Cir. 2000); *In re Dembiczak*, 50 U.S.P.Q.2d. 1614, 1617 (Fed. Cir. 1999). Broad conclusory statements regarding the teaching of multiple references, standing alone, are not evidence. *Id.*

In order to reject under 35 U.S.C. § 103(a), therefore, the Examiner must provide a proper motivation for combining or modifying the references. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1457-1458 (Fed. Cir. 1998); M.P.E.P. § 2142. The Examiner's motivation for modifying *Burgess* to determine if a specified condition register field is used to store a branch condition of the conditional branch instruction is "so the condition data may be used later in the program." Paper No. 3, page 5.

There is no motivation to combine *Burgess* and *Intel* as there is no suggestion or motivation in either *Burgess* or in *Intel*, or in their combination, or in the knowledge of those ordinarily skilled in the art to combine the teaching of a method of loading a particular block of instructions into the instruction cache involving repetitively mis-predicting a branch instruction in a loop, as taught in *Burgess*, with a teaching of storing condition codes in a register, as taught in *Intel*. *Burgess* teaches:

> A *method of loading a particular block of instructions into the instruction cache (14) of a Harvard architecture data processor (10) involves repetitively mis-predicting a branch instruction in a loop*. The branch instruction is conditioned upon an instruction whose execution is contrived to output a sequential fetch address. However, the instruction's result is not available until after the branch instruction begins executing. Therefore, the data processor speculatively executes or predicts the branch instruction. In this case, the branch instruction predicts that it will branch to the particular block of instructions. The data processor then loads the instructions into its instruction cache. Later, the data processor determines that it mis-predicted the branch instruction, returning to the loop for another iteration. Abstract.

Thus, *Burgess* teaches a method of loading a particular block of instructions into an instruction cache which involves repetitively mis-predicting a branch instruction in a loop.

> *Intel* teaches:
>
> Condition codes, (e.g., carry, assign, overflow) and mode bits are kept in a 32-bit register named EFLAGS. Figure 3-9 defines the bits within this register. 3-13.

Thus, *Intel* teaches storing condition codes in a register.

The Examiner has not shown why a reference that teaches a method of loading a particular block of instructions into an instruction cache which involves repetitively mis-predicting a branch instruction in a loop, as taught in *Burgess*, should be combined with a reference that teaches storing condition codes in a register, as taught in *Intel,* from either the nature of the problem to be solved, the teachings of the prior art, or in the knowledge of persons of ordinary skill in the art. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998). The Examiner must submit **objective evidence** and not rely on his own subjective opinion in support of combining the reference that teaches a method of loading a particular block of instructions into an instruction cache which involves repetitively mis-predicting a branch instruction in a loop with a reference that teaches storing condition codes in a register. *In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002). Therefore, the Examiner has not presented a *prima facie* case of obviousness for rejecting claims 1-14.

As stated above, the Examiner's motivation for modifying *Burgess* to determine if a specified condition register field is used to store a branch condition of the conditional branch instruction is so that the condition data may be used later in the program. The Examiner has not shown why *Burgess* should be modified to determine if a specified condition register field is used to store a branch condition of the conditional branch instruction from either the nature of the problem to be solved, the teachings of the prior art or in the knowledge of persons of ordinary skill in the art. *In re Rouffet*, 47 U.S.P.Q.2d 1453, 1458 (Fed. Cir. 1998). Further, the Examiner has not shown why *Burgess* should be modified so that condition data may be used later in a program from either the nature of the problem to be solved, the teachings of the prior art or in the knowledge of persons of ordinary skill in the art. *Ibid.* The Examiner must submit **objective evidence** and not rely on his own subjective opinion in support of modifying *Burgess* to determine if a specified condition register field is used to store a branch condition of the conditional branch instruction. *In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002). Further, the Examiner must submit **objective evidence** and not rely on his own subjective opinion in support of modifying *Burgess* so that the condition data may be used later in the program. *Id.* Accordingly, the Examiner has not presented a *prima facie* case of obviousness for rejecting claims 1-14.

B. Burgess and Intel, taken singly or in combination, do not teach or suggest the following limitations

Burgess and *Intel*, taken singly or in combination, do not teach or suggest "*determining if a specified condition register field is used to store a branch condition of the conditional branch instruction*" as recited in claim 1 and similarly in claim 8. The Examiner directs Applicant's attention to pages 3-13 to 3-15 of *Intel* as teaching the above-cited claim limitation. Paper No. 3, page 5. Instead, Intel teaches:

> Condition codes (e.g., carry, assign, overflow) and mode bits are kept in a 32-bit register named EFLAGS. Figure 3-9 defines the bits within this register. 3-13.

Thus, *Intel* teaches storing condition codes in a register. However, this language does not teach a condition register field used to store a *branch* condition. Further,

this language does not teach *determining if* a specified *condition register field is used to store a branch condition of a conditional branch instructions*. Therefore, the Examiner has not provided a *prima facie* case of obviousness for rejecting claims 1 and 8. M.P.E.P. § 2143.

Burgess and *Intel*, taken singly or in combination, do not teach or suggest "*providing a software branch prediction of the conditional branch instruction as a function of the determination if the specified condition register field is used to store the branch condition of the conditional branch instruction*" as recited in claim 1 and similarly in claim 8. The Examiner directs Applicant's attention to column 4, lines 11-31 of *Burgess* as teaching the above-cited claim limitation. Paper No. 3, page 4. Instead, *Burgess* teaches:

> Data processor 10 also incorporates a static branch prediction methodology to ensure a constant supply of instructions to its various execution units 20, 22, 24, 26, 28 and 30. *According to a static branch prediction methodology, if data processor 10 has not yet determined the condition upon which a branch instruction is based, then it assumes either that the branch will be taken or that the branch will not be taken depending upon one or more bits in the branch instruction itself. Data processor 10 then fetches instructions at the taken or not-taken address, as appropriate, before it actually calculates the condition upon which the branch instruction is based.* Later, data processor 10 calculates the condition and, if mispredicted, returns to the fetch address not previously selected. Otherwise, data processor continues executing instructions along the predicted path. *At program compilation, each branch instruction is predicted as taken or not-taken depending upon the statistical likelihood that the branch is predominantly taken or is predominantly not taken when executed. One or more bits in the instruction itself indicates to data processor 10 whether the branch instruction should be taken or not-taken.* Column 4, lines 11-31.

Thus, *Burgess* teaches that at program compilation, each branch instruction is predicted as taken or not-taken depending upon the statistical likelihood that the branch is predominately taken or is predominantly not taken when executed. *Burgess* further teaches that one or more bits in the instruction itself indicates to the data processor whether the branch instruction should be taken or not-taken. This language does not teach *providing a software branch prediction of a conditional branch instruction*. Further, this language does not teach providing a

software branch prediction of a conditional branch instructions *as a function of the determination if a specified condition register field is used to store a branch condition of a conditional branch instructions*. Further, *Burgess* remains silent as to a *condition register field*. Further, *Burgess* is silent regarding a condition register field *used to store a branch condition of the conditional branch instruction*. Therefore, the Examiner has not provided a *prima facie* case of obviousness for rejecting claims 1 and 8. M.P.E.P. § 2143.

For at least the above reasons, claims 1 and 8 are patentable over *Burgess* in view of *Intel*. Claims 2-7 and 9-14 each recite combinations of features including the above combinations, and thus are patentable for at least the above reasons as well. Claims 2-7 and 9-14 recite additional features, which, in combination with the features of the claims upon which they depend, are patentable over *Burgess* in view of *Intel*.

For example, *Burgess* and *Intel*, taken singly or in combination, do not teach or suggest "wherein the *software branch prediction predicts that the conditional branch instruction will be taken if the specified condition register field is used to store the branch condition of the conditional branch instruction*" as recited in claim 2 and similarly in claim 9. Further, *Burgess* and *Intel*, taken singly or in combination, do not teach or suggest "wherein the *software branch prediction predicts that the conditional branch instruction will be not taken if the specified condition register field is not used to store the branch condition of the conditional branch instruction*" as recited in claim 3 and similarly in claim 10. Further, *Burgess* and *Intel*, taken singly or in combination, do not teach or suggest "wherein the *software branch prediction predicts that the conditional branch instruction will be not taken if the specified condition register field used to store the branch condition of the conditional branch instruction*" as recited in claim 4 and similarly in claim 11. Further, *Burgess* and *Intel*, taken singly or in combination, do not teach or suggest "wherein the *software branch condition predicts that the conditional branch instruction will be taken if the specified condition register field is not used to store the branch condition of the conditional branch instruction*" as recited in claim 5 and similarly in claim 12.

The Examiner directs Applicant's attention to column 4, lines 11-31 of *Burgess* as teaching the above-cited claim limitations. Paper No. 3, pages 5-6. Instead, as stated above, *Burgess* teaches that at program compilation, each branch instruction is predicted as taken or not-taken, depending upon the physical likelihood that the branch is predominantly taken or is predominantly not taken when executed. *Burgess* further teaches that one or more bits in the instruction itself indicates to the data processor whether the branch instruction should be taken or not-taken. This language does not teach a *condition register field.* Further, this language does not teach a *condition register field used to store or not store a branch condition of a conditional branch instruction.* Further, this language does not teach a *software branch prediction that predicts that the conditional branch instruction will be taken or not taken if a condition register field is used or not used to store the branch condition of a conditional branch instruction.* Therefore, the Examiner has not presented a *prima facie* case of obviousness for rejecting claims 2-5 and 9-12. MPEP § 2143.

Burgess and *Intel*, taken singly or in combination, do not teach or suggest "wherein the *specified condition register field is N, where N is an integer*" as recited in claim 6 and similarly in claim 13. Further, *Burgess* and *Intel*, taken singly or in combination, do not teach or suggest "wherein the *specified condition register field is a multiple of N*" as recited in claim 7 and similarly in claim 14. The Examiner states that *Burgess* expressly teaches the above-cited claim limitation without citing any particular passage in *Burgess* as support for the Examiner's assertion. Paper No. 3, page 6. Upon review of *Burgess*, Applicant does not locate any passage that taught a condition register field used to store a branch condition of a conditional branch instruction. Hence, *Burgess* does not teach *where a specified condition register field is N, where N is an integer.* Neither does *Burgess* teach *where a specified condition register field is a multiple of N.* The Examiner further directs Applicant's attention to pages 3-13 to 3-15 of *Intel* as teaching the above-cited claim limitation. Paper No. 3, page 6. Instead, as stated above, *Intel* teaches storing condition codes in a register. However, this language does not teach a condition register field used to store a branch condition

of a conditional branch instructions. Hence, this language does not teach *where a specified condition register field is N, where N is an integer.* Neither does this language teach *where a specified condition register field is a multiple of N.* Therefore, the Examiner has not presented a *prima facie* case of obviousness for rejecting claims 6-7 and 13-14. M.P.E.P. §2143.

In connection with the Examiner's assertion that *Intel* teaches the above-cited claim limitations in claims 6-7 and 13-14, the Examiner states that "changing the field the condition is stored in is just shifting the location of the part. *See In re Japikse*, 181 F.2d 1019, 86 U.S.P.Q. 70 (C.C.P.A. 1950)." Paper No. 3, page 6. Applicant notes that *In re Japikse*, upon which the Examiner relies, precedes *Graham vs. John Deere Co.*, 383 U.S. 1, 148 U.S.P.Q. 459 (1966). Thus, to the extent that *In re Japikse* anticipates the factual analysis mandated in *Graham*, they are merely cumulative of *Graham*. Conversely, to the extent they are contrary to the factual inquiry mandated by *Graham*, it must be deemed overruled by *Graham*.

Furthermore, *In re Japikse* considers questions involving jurisdiction and the operativeness of the device disclosed in the Cannon patent. *In re Japikse*, 181 F.2d 1019, 1021-1022 (C.C.P.A. 1950). Applicant has been unable to locate in *In re Japikse* where *In re Japikse* stands for the proposition that changing the field the condition is stored in is just shifting the location of the part. Applicant respectfully requests the Examiner to particularly point out in *In re Japiske* where *In re Japikse* stands for the proposition asserted by the Examiner pursuant to 37 C.F.R. § 1.104(c)(2). Therefore, *In re Japikse* does not support the assertion that *Intel* teaches the claimed invention of claims 6-7 and 13-14.

C. Conclusion regarding 35 U.S.C. § 103 rejections

As a result of the foregoing, Applicant respectfully asserts that there are numerous claim limitations not taught or suggested in the cited prior art, and thus the Examiner has not presented a *prima facie* case of obviousness for rejecting claims 1-14 in view of the cited prior art.

It is noted that limitations are italicized only for emphasis. Limitations that are italicized are not meant to imply that only those limitations are not taught or suggested in the cited prior art.

V. INTERVIEW SUMMARY:

Applicant thanks the Examiner for discussing the office action and in particular claim 39 with Applicant on July 1, 2003.

VI. CONCLUSION:

As a result of the foregoing, it is asserted by Applicant that claims 1-14 and 39-40 in the Application are in condition for allowance, and Applicant respectfully requests an allowance of such claims. Applicant respectfully requests that the Examiner call Applicant's attorney at the below listed number if the Examiner believes that such a discussion would be helpful in resolving any remaining issues.

Respectfully submitted,

WINSTEAD SECHREST & MINICK P.C.

Attorneys for Applicant

By:_____

Robert A. Voigt, Jr.
Reg. No. 47,159
Kelly K. Kordzik
Reg. No. 36,571

P.O. Box 50784
1201 Main Street
Dallas, Texas 75250-0784
(512) 370-2832

AUSTIN_1\220504\2
7047-P315US